Constructing optimal designs for order-of-addition experiments using a hybrid algorithm

Dongying Wang[a], Sumin Wang[b,*]

[a]*School of Statistics, Jilin University of Finance and Economics, Changchun, Jilin 130117, China*
[b]*Center for Combinatorics, LPMC & KLMDASR, Nankai University, Tianjin 300071, China*

Abstract

For order-of-addition experiments, the response is affected by the addition order of the experimental materials. Consequently, the main interest focuses on creating a predictive model and an optimal design for optimizing the response. Van Nostrand (1995) proposed the pairwise-order (PWO) model for detecting PWO effects. Under the PWO model, the full PWO design is optimal under various criteria but is often unaffordable because of the large run size. In this paper, we consider the D-, A- and $M.S.$-optimal fractional PWO designs. We first present some results on information matrices. Then, a flexible and efficient algorithm is given for generating these optimal PWO designs. Numerical simulation shows that the generated design has an appealing efficiency in comparison with the full PWO design, though with only a small fraction of runs. Several comparisons with existing designs illustrate that the generated designs achieve better efficiencies, and the best PWO designs and some selected 100% efficient PWO designs generated by the new algorithm are reported.

Keywords: Pairwise-order model, D-optimal, A-optimal, $M.S.$-optimal, particle swarm optimization, Fedorov exchange algorithm.

2010 MSC: 62K15, 62K99

1. Introduction

In some specific experiments, such as a chemical experiment with a number of reactants that are added into an apparatus sequentially rather than simultaneously, different orders of adding the components involved in the system yield different responses. Therefore, researchers are more interested in how the addition sequence of reactants affects the response. Experiments with this feature are referred to as order-of-addition (OofA) experiments and are widely applied to chemical-related areas and food industries, as well as biochemistry and measurement processes. The earliest research on OofA experiments can perhaps be traced back to the study of a lady tasting tea in Fisher (1937). Another study appeared in Fuleki and Francis (1968) that evaluated an experiment

* Corresponding author
Email address: wangsm088@nankai.edu.cn (Sumin Wang)

for extracting anthocyanins from cranberries. During the past decades, the approach of OofA experiments has been proposed in many practical studies; for example, see the references Jourdain et al. (2009), Karim, Mccormick, and Kappagoda (2000), Olsen et al. (1994) and so on.

For the objective of optimizing and predicting the response, a statistical model and an optimal design are created for the OofA experiment. The idea of pairwise-order (PWO) modeling and designing the OofA experiment has been presented in Van Nostrand (1995). Recently, Voelkel (2019) proposed a number of design criteria. Voelkel (2019) provided theoretical results on the full PWO design and construction of the optimal PWO design, which has the same correlation structure as the full PWO design, namely, the same information matrix. A recent review on OofA experiments and PWO models can be found in Lin and Peng (2019).

In fact, the PWO model is also a regression model. Then, a family of criteria can be applied to find optimal designs under the PWO model, such as D-, A- and $M.S.$-optimal designs. The optimality proof indicates that a full PWO design with $m!$ distinct permutations of components is D-, A- and $M.S.$-optimal, but the run size is extremely large. Taking $m = 10$ as an example, there are $m! = 3628800$ distinct permutations. Consequently, over three million runs of experiments should be implemented, which is impractical. Therefore, fractions of full PWO designs with a smaller number of runs are preferable.

Recently, four kinds of fractional PWO designs have been studied. Peng, Mukerjee, and Lin (2019) introduced a method for constructing optimal PWO designs. This method limits the run size to $m!/r!$ $(2 \leq r \leq m/2)$, which is also too many for experimenters to afford. For instance, if $m = 10$, the method needs at least 30240 runs to be implemented. Yang, Sun, and Xu (2021) and Zhao, Lin, and Liu (2022) provided construction methods based on an orthogonal array, the resulting designs are component orthogonal arrays (COAs) and OofA orthogonal arrays (OofA-OAs), in which the run size is also inflexible. Zhao, Lin and Liu (2021) provided a minimal-point design with $m(m-1)/2+1$ runs. The run size is small, but the efficiency is relatively low. However, theoretical constructions of these fractional PWO designs are highly dependent on run size. Winker, Chen, and Lin (2020) applied the threshold accepting algorithm to construct the optimal designs (D-efficiency for application) based on the pairwise-order (PWO) model and the tapered PWO model, the designs obtained by threshold accepting algorithm for $4 \leq m \leq 30$ with $n = m(m-1)/2+1, m(m-1)+1, 3m(m-1)/2+1$, respectively, are provided for practical uses. The present paper also provides a computer algorithm to construct the PWO design with a flexible run size, and D-, A-, and $M.S.$-optimal PWO designs can be constructed using the proposed algorithm. When compared with the full PWO designs, the constructed designs possess high efficiencies.

This paper is organized as follows. We first introduce the PWO model in Section 2. Section 3 gives a review of Fedorov's exchange algorithm for constructing the D-optimal designs. Then, this algorithm is modified and extended for constructing A- and $M.S.$-optimal designs. Some theoretical results on the information matrix and algorithm are also provided in Section 3. In

Section 4, based on the exchange algorithm and particle swarm optimization (PSO) algorithm, a novel hybrid algorithm is proposed to achieve D-, A- and $M.S.$-optimal PWO designs. Some numerical results are given in Section 5. Finally, concluding remarks are provided in Section 6.

2. Model specification

Now, we introduce the Van Nostrand PWO model. Suppose there are m components denoted as $1, \ldots, m$. Any treatment in the OofA experiment corresponds to a permutation of $1, \ldots, m$, denoted as $\boldsymbol{\alpha}$, and the first-order PWO model can be expressed as

$$\tau(\boldsymbol{\alpha}) = \beta_0 + \sum_{1 \le j < k \le m} z_{jk}(\boldsymbol{\alpha}) \beta_{jk},$$

where each $z_{jk}(\boldsymbol{\alpha})$ is a PWO indicator between j and k,

$$z_{jk}(\boldsymbol{\alpha}) = \begin{cases} 1 & \text{if } j \text{ precedes } k \text{ in } \boldsymbol{\alpha}, \\ -1 & \text{if } k \text{ precedes } j \text{ in } \boldsymbol{\alpha}. \end{cases} \tag{1}$$

For an n-point PWO design, let Y be the n-dimensional response vector, Z be the design matrix with $\binom{m}{2}$ columns corresponding to PWO indicators $z_{12}, z_{13}, \ldots, z_{(m-1)m}$, and $\boldsymbol{\beta} = (\beta_{12}, \beta_{13}, \ldots, \beta_{(m-1)m})'$, where $'$ denotes the transpose. Then, the first-order PWO model can be written as

$$Y = \mathbf{1}\beta_0 + Z\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{2}$$

or

$$Y = X\widetilde{\boldsymbol{\beta}} + \boldsymbol{\epsilon}, \tag{3}$$

where $X = (\mathbf{1}\ Z)_{n \times p}$ with $p = \binom{m}{2} + 1$ is the model matrix, and $\widetilde{\boldsymbol{\beta}} = (\beta_0, \boldsymbol{\beta}')'$ represents the parameter of interest. Mee (2020) extended the PWO model to the high-order case. Here, we only consider a first-order PWO model. The proposed algorithms also apply to a higher-order PWO model.

Furthermore, we refer to $\widetilde{M} = X'X/n$ as the information matrix of an n-point PWO design. Under the PWO model (3), the variance-covariance matrix of the least squares estimator of $\boldsymbol{\beta}$ is proportional to $\widetilde{M}$. Hence, it is desirable to maximize the matrix $\widetilde{M}$ under some criteria. The popular criteria include the D-criterion $det(\widetilde{M})^{1/p}$, the A-criterion $tr(\widetilde{M}^{-1})$, the $M.S.$-criterion $tr(\widetilde{M}^2)$ (see the reference Atwood 1969). Note that $tr(\widetilde{M}^{-1})$ is interpreted as $+\infty$ for singular $X'X$. Let X_f be the full PWO design and the corresponding information matrix be $\widetilde{M}_f = X_f'X_f/n$. For clarity, we take $m = 3$ as an example to illustrate the characteristics of the full PWO design under D-, A- and $M.S.$-criteria. The levels of PWO factors in the full PWO design with 3 components are as follows.

Table 1. Full PWO design with 3 components

Run	Order-of-Addition	z_{12}, z_{13}, z_{23}
1	1 2 3	$1, \ 1, \ 1$
2	1 3 2	$1, \ 1, -1$
3	2 1 3	$-1, \ 1, \ 1$
4	2 3 1	$-1, -1, \ 1$
5	3 1 2	$1, -1, -1$
6	3 2 1	$-1, -1, -1$

From this, we obtain

$$X_f = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 \\ 1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & -1 \end{pmatrix}, \ \widetilde{M}_f = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1/3 & -1/3 \\ 0 & 1/3 & 1 & 1/3 \\ 0 & -1/3 & 1/3 & 1 \end{pmatrix},$$

and $det(\widetilde{M}_f) = 16/27$, $tr(\widetilde{M}_f^{-1}) = 11/2$ and $tr(\widetilde{M}_f^2) = 14/3$.

3. Exchange algorithms for constructing *D*-, *A*-, and *M.S.*-optimal designs

Theoretical constructions on optimal designs are always complicated; hence, computer algorithms are applied for constructing approximate and exact optimal designs in the literature. Exchange algorithm is one of the popular computer algorithms for constructing optimal designs for the cases with the design points being selected from a finite design space. Fedorov (1972) first proposed an exchange algorithm for generating D-optimal designs. This algorithm chooses n points to include in the design from a finite set of possible points called candidate points, and it starts with nonsingular n-point designs and then adds and deletes one observation in order to achieve increases in the determinant. After that some improved implementations are proposed based upon Fedorov's exchange algorithm, such as the Kiefer round-off algorithm, the Mitchell algorithm, the Wan Schalkwyk algorithm, the combined Fedorov, the Wynn-Mitchell algorithm and so on; see the references Mitchell (1992), Nguyen and Miller (1992).

3.1. The single-point exchange procedure

Consider an n-point design $D = \{\alpha_i\}_{i=1}^n$ under model (3), with a corresponding model matrix $X = (x_1, \ldots, x_n)'$. If $\widetilde{M} = X'X/n$, then the D-, A- and $M.S.$-criteria maximize $det(\widetilde{M})^{1/p}$, $-tr(\widetilde{M}^{-1})$ and $-tr(\widetilde{M}^2)$ respectively, which are equivalent to maximizing $\phi(D) = det(M)$, $-tr(M^{-1})$ and $-tr(M^2)$ respectively, where $M = X'X = \sum_{i=1}^n x_i x_i'$.

Inspired by Fedorov's exchange algorithm, we develop a new exchange algorithm for generating D-, A- and $M.S.$-optimal designs simultaneously. This algorithm is realized by multiple iterations of the single-point exchange procedure which works as follows.

Single-point exchange procedure:

Let X be the model matrix of the original design and $M = X'X$,

(1) Find a vector x among the vectors of the complementary design such that $u(x)$ is maximum and add x to the current n-point design;

(2) Find a vector x_i among the $n+1$ vectors of the current $n+1$-point design such that $v(x_i)$ is minimum and remove x_i.

When use the single-point exchange procedure for generating the D-, A- and $M.S.$-optimal designs, the objective functions are denoted as $u_*(x)$ and $v_*(x_i)$ with $* = D, A, M.S.$ and defined as bellow:

$$u_D(x) = x'M^{-1}x, v_D(x_i) = x_i'M_x^{-1}x_i; \tag{4}$$

$$u_A(x) = \frac{x'M^{-2}x}{1 + x'M^{-1}x}, v_A(x) = \frac{x_i'M_x^{-2}x_i}{1 - x_i'M_x^{-1}x_i}; \tag{5}$$

$$u_{M.S.}(x) = -x'Mx, v_{M.S.}(x_i) = -x_i'M_xx_i; \tag{6}$$

where $M = \sum_{i=1}^{n} x_ix_i'$ is the moment matrix of the current design and M is updated to $M_x = M + xx'$ when a candidate point from the complementary design is added to the current design. Here, the complementary design consists of all candidate points from the design space except for the n points of the current design.

Theorem 3.1. *For D-, A-, and $M.S.$-criteria which maximize $\phi(D) = det(M), -tr(M^{-1})$ and $-tr(M^2)$ respectively, the design generated by the single-point procedure with $u(x)$ and $v(x_i)$ defined as equations (4)-(6) leads to no decrease in $\phi(D)$.*

The proof of this theorem uses some matrix theories, and we present it in the appendix. This result implies that exchange algorithm will return local D-, A- and $M.S.$-optimal designs over multiple iterations of the single-point exchange procedure.

3.2. The technique for avoiding the singularity of the matrix for the exchange algorithm

For generating optimal design using a computer search algorithm, the solution is often trapped into the local optimal design. Thus random exchange method is always used to avoid this drawback. For constructing D-, A- and $M.S.$-optimal designs using a computer search algorithm, a random selected initial design possibly corresponds to a singular moment matrix, especially for the case

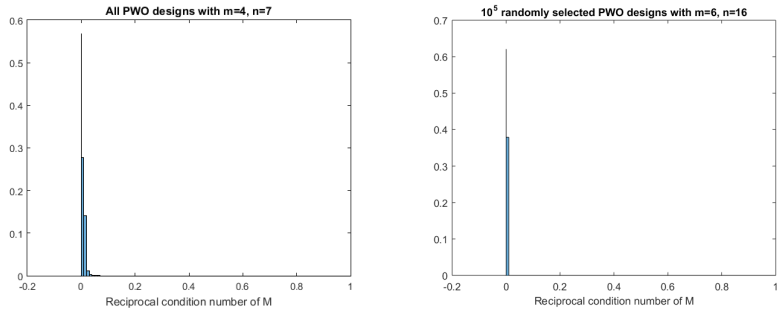


Figure 1: Distributions of the reciprocal condition numbers of matrix M for all PWO designs with $m = 4, n = 7$ and 10^5 randomly selected PWO designs with $m = 5, n = 11$.

with a rather small number of n, and computation problem then arises. Taking the case with $n = \binom{m}{2} + 1 (m = 4, 5)$ as an example. Among all $\binom{m!}{n}$ options of n-point design, a large proportion of them correspond to a badly conditioned matrix M. As shown in Figure 1, all the reciprocal condition numbers are near 0, and the reciprocal condition number below 10^{-12} is counted in the first bin of each histogram with a probability exceeding 50%.

A random selected initial design will return a computationally singular matrix M with a large probability. For this reason, we address the issue of avoiding singularities of M and M_x in the single-point exchange algorithm. Two types of techniques are provided regarding this issue. The first technique is to start with a nonsingular design instead of starting with a randomly selected design.

Remark 3.2. *If the initial design has nonsingular moment matrix, then by* $M_x = |M|(1+x'M^{-1}x)$ *and* $M_x - x_i x_i' = |M_x|(1 - x_i' M_x^{-1} x_i)$, *where* $x_i' M_x^{-1} x_i \leq x' M_x^{-1} x = \frac{x' M^{-1} x}{1+x' M^{-1} x} < 1$, *both* M_x *and* $M_x - x_i x_i'$ *are nonsingular matrices during each iteration of the single-point exchange algorithm which is performed recursively.*

This technique is practical since a nonsingular initial design with n points can be obtained by appending $n - \binom{m}{2} - 1$ randomly selected distinct points to the minimal-point design provided in Zhao, Lin, and Liu (2021). However, in the hybrid algorithm, the design is updated via both the single-point exchange procedure and some random exchange procedure.

The second technique is inspired by the DETMAX algorithm in Mitchell (2000), a specified nonsingular matrix multiplied by a very small positive parameter θ is added to matrix M or M_x. Taking M as an example, we do not consider M^{-1} directly, but instead attempt to calculate $(M + \theta(X_f' X_f / N_f))^{-1}$, where N_f is the number of candidate points, and X_f is the model matrix of the full design composed of all N_f candidate points. Then, one technique that we can use to avoid the singularity of the matrix is as follows.

Remark 3.3. *To avoid singularity,* $x'(M + \theta(X_f' X_f / N_f))^{-1} x$ *and* $x_i'(M_x + \theta(X_f' X_f / N_f))^{-1} x_i$ *are maximized and minimized in the single-point exchange algorithm with* $u(x)$ *and* $v(x_i)$ *being defined*

as equations (4) and (5). The degree of error involved in considering these alternative matrices is less than θ.

To appreciate the degree of error involved in considering the alternative matrix, one can make the following calculations. Let

$$f(\theta) = x'(M + \theta(X_f'X_f/N_f))^{-1}x.$$

Then, extend $f(\theta)$ in a Taylor series about $\theta = 0$ to obtain the linear approximation:

$$\begin{aligned} f(\theta) &\cong f(0) + \theta\left(\frac{df(\theta)}{d\theta}\right)\Big|_{\theta=0} \\ &= x'M^{-1}x - \theta\big(x'(M + \theta(X_f'X_f/N_f))^{-1}(X_f'X_f)/N_f(M + \theta(X_f'X_f/N_f))^{-1}x\big)\big|_{\theta=0} \\ &= x'M^{-1}x - \theta x'M^{-1}X_f'X_fM^{-1}x/N_f. \end{aligned}$$

For small θ, the error in considering $x'(M + \theta(X_f'X_f/N_f))^{-1}x$ instead of $x'M^{-1}x$ is nearly $\theta x'M^{-1}X_f'X_fM^{-1}x/N_f$. In the proposed algorithm, the value of θ is set at 0.005, which is found to be quite satisfactory in simulations. This choice based on run size of the full PWO is sufficiently large such that $x'M^{-1}X_f'X_fM^{-1}x/N_f < 1$, and the error will be less than 0.5%.

Note that in this paper, we adopt the technique described in Remark 3.3 to avoid the singularity of the matrix.

3.3. The performance of the exchange algorithm

Now we discuss the performance of the exchange algorithm. The single-point exchange procedure is performed recursively, and the D-, A- and $M.S.$-efficiencies of the generated designs are calculated. For brevity, the cases with $m = 4, 5, 6, 7$ components are considered and the run sizes are fixed at $n = m(m-1)$. The following Figure 2 shows that the efficiencies are deeply increased in former iterations but then stabilized at slows on one value as the number of iterations increased. Therefore, the exchange algorithm yields locally optimal designs that approximate a global optimal design in a reasonable number of iterations.

To illustrate the performance of the exchange algorithm for constructing D-, A- and $M.S.$-optimal designs, 1000 designs are generated by the exchange algorithm with respective to each pair of the objective functions defined in equations (4)-(6). The initial designs are randomly selected. We list the minimum, average and maximum efficiencies of the generated designs in Table 2. Obviously, the generated designs are largely depended on the initial designs, most of them are locally optimal designs and some of them even have lower efficiencies than 80%, see the numbers in a bold font. Thus, in the next section, we proposed a more robust hybrid algorithm which combines the exchange algorithm and the particle swarm algorithm to produce approximate optimal designs with higher efficiency than the designs generated by exchange algorithm.

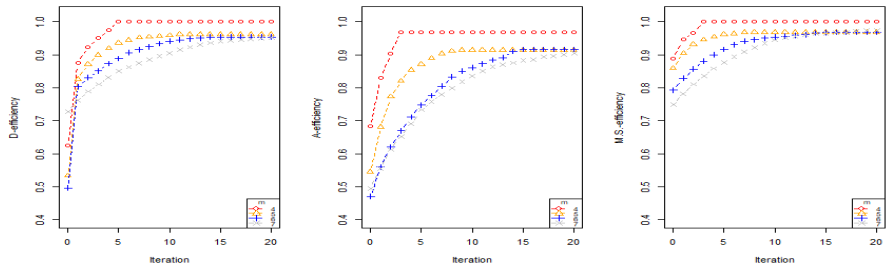


Figure 2: D-, A- and $M.S.$-efficiencies of PWO designs generated initial design, and each graph contains four lines corresponding to PWO designs with $m = 4, 5, 6, 7$ components and $n = m(m-1)$ runs, respectively.

Table 2. Efficiencies of 1000 designs generated by the exchange algorithm

		D-efficiency			A-efficiency			$M.S.$-efficiency		
m	Runs	Min	Ave	Max	Min	Ave	Max	Min	Ave	Max
4	12	97.4%	99.8%	100%	**32.1%**	**65.3%**	92.4%	**76.3%**	97.7%	100%
5	20	94.2%	96.0%	97.0%	**19.5%**	**51.7%**	**73.9%**	**74.4%**	96.5%	98.2%
6	30	93.8%	95.8%	97.1%	**19.1%**	**48.0%**	**69.5%**	95.2%	96.8%	98.1%
7	42	93.7%	95.3%	96.7%	**33.4%**	**47.7%**	**63.4%**	95.9%	97.1%	98.0%

4. Constructions on D-, A- and $M.S.$-optimal PWO designs using a hybrid algorithm combining the exchange algorithm and PSO algorithm

Before introducing the new algorithm, we add some details of the PSO algorithm. PSO is a population-based stochastic algorithm for optimization. Each population member is described as a particle that moves around a search space testing new criterion values. All particles survive from the beginning of a trial until the end, and their interactions result in iterative improvement of the quality of the problem solutions over time. The most common type of implementation defines the particles' behavior as adjusting toward each of its personal best position(local-best) and global-best position so that its trajectory shifts to new regions of the search space and the particles gradually cluster around the optima. For applications to find optimal experimental designs, a particularly challenging task is to redefine the particle designs' movement toward its personal local-best design and global-best design. A review of some recent applications of PSO and its variants to tackle various types of efficient experimental design is Chen, Chen, and Wang (2022). Since finding optimal PWO designs for OofA experiment is to solve a discrete optimization problem, we utilize a update procedure for the particle designs that is similar to the modified PSO algorithms in Chen et al. (2014) and Phoa et al. (2016). Each particle design relates to its personal local-best design which is derived by exchange procedures starting from itself. During each iteration, the current particle design is adjusted toward its personal local-best design as well as the global-best design by exchanging points with each other.

Now, we introduce a new hybrid algorithm called *Ex-PSO* algorithm, which combining the single-point exchange algorithm and PSO algorithm for generating D-, A-, and $M.S.$-optimal designs. The single-point exchange algorithm is used for generating the local-best design with respect to each particle design. The PSO algorithm ensure the particle designs gradually cluster around the optimal PWO design. To avoid singularity, the technique proposed in Remark 3.3 is used; hence, a parameter θ with a small value is involved in this algorithm.

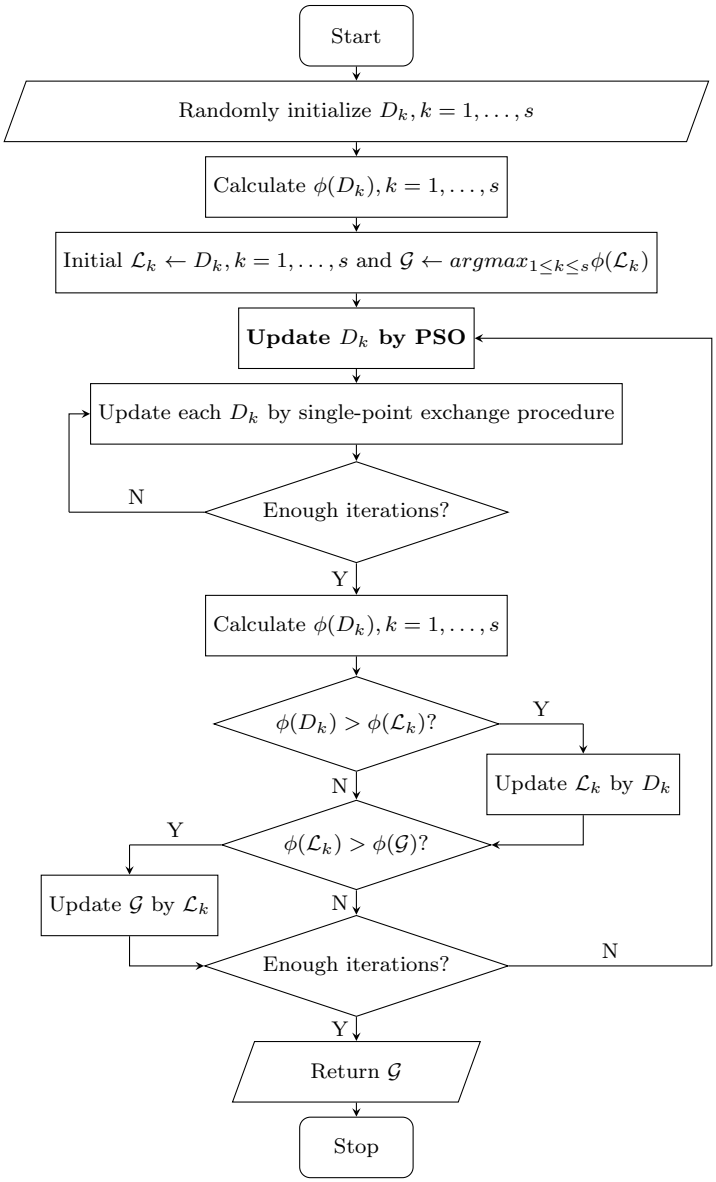
Since the *Ex-PSO* algorithm involves a set of parameters denoted as $s, t_{ex}, t_{pso}, \theta, c_1, c_2$, we also refer to it as Ex-PSO$(m, n; s, t_{ex}, t_{pso}, \theta, c_1, c_2)$ for generating optimal PWO design with m components and n runs. For clarity, we create a programming chart to illustrate the steps of Ex-PSO$(m, n; s, t_{ex}, t_{pso}, \theta, c_1, c_2)$. Further, we explain the optimization process and the uses of these parameters as follows. Denote $\mathcal{L}_k$s and $\mathcal{G}$ as the local-best designs and the global-best design respectively. These designs are updated during each iteration of the *Ex-PSO* algorithm. Each local-best design $\mathcal{L}_k$ is derived from the current particle design D_k via a fixed number of iterations of the single-point exchange procedure, denoted as t_{ex}. In addition, the global-best design $\mathcal{G}$ is the optimal local-best design that maximizes $\phi(\mathcal{L}_k)$. And the number of iterations of the PSO algorithm is denoted as t_{pso}. Meanwhile, two parameters are used to control the PSO behavior of the *Ex-PSO* algorithm: c_1 and c_2, which account for the velocities at which each current design drifts toward the corresponding local-best and global-best design. More specifically, during each iteration of the PSO algorithm, we randomly exchange c_1 points from the difference set $D_k \setminus \mathcal{L}_k$ with c_1 points from $\mathcal{L}_k \setminus D_k$ and then randomly exchange c_2 points from the difference set $D_k \setminus \mathcal{G}$ with c_2 points from $\mathcal{G} \setminus D_k$. This procedure corresponds to the "Update D_k by PSO" box in the programming chart.

Finally, we note that the *Ex-PSO* algorithm is implemented in MATLAB running on Intel(R) Core(TM) i7-8550U GHz with 8 GB Memory. Take the case of $m = 6, n = 16$ for example, it takes 30.42 seconds for running *Ex-PSO*$(6, 16; 10, 20, 100, 0.005, 1, 1)$.

5. Numerical simulations

In this section, we illustrate the performances of the obtained designs constructed by the *Ex-PSO* algorithm. For brevity, the generated designs are denoted as *Ex-PSO-D*, *Ex-PSO-A* and *Ex-PSO-M.S.* designs that respectively correspond to the objective functions (4)-(6) which are considered in the exchange algorithm. Numerical simulations show that these designs are powerful for fitting PWO models in terms of the D-, A- and $M.S.$-efficiencies. The efficiencies are derived from comparison with the full PWO design, since the information matrix of the full PWO design has been proven to be universally optimal. Therefore, we have the D-, A- and $M.S.$-efficiencies that calculate $\frac{det(\widetilde{M})^{1/p}}{det(\widetilde{M}_f)^{1/p}}$, $\frac{tr(\widetilde{M}_f^{-1})}{tr(\widetilde{M}^{-1})}$ and $\frac{tr(\widetilde{M}_f^2)}{tr(\widetilde{M}^2)}$ respectively, where $\widetilde{M}$ and $\widetilde{M}_f$ are the information matrices of the obtained design and full PWO design respectively, and p is the number of the columns of the model matrix X.

***Ex-PSO* Algorithm**: Ex-PSO$(m, n; s, t_{ex}, t_{pso}, \theta, c_1, c_2)$



Clearly, the number of PSO particles (s), the maximum iteration counts of single-point exchange algorithm and PSO algorithm (t_{ex}, t_{pso}) and the numbers of pairs of exchanging points with which each particle design drifts toward the local-best and global-best design (c_1, c_2), control the optimization process of *Ex-PSO* algorithm. It seems reasonable that these parameters should be larger for larger problems. In our test of searching for optimal PWO designs with $m = 4, 5, 6, 7$ components, we recommend that these parameters to set at $s = 10$, $t_{ex} = 20$, $t_{pso} = 100$, $c_1 = c_2 = 1$. Furthermore, we recommend to set the maximum iteration counts of exchange algorithm and PSO at $t_{ex} = 20$ and $t_{pso} = 100$ respectively, which achieves high computational efficiency. Further, to demonstrate the performance of such a set of parameters, we randomly run the algorithm Ex-PSO$(m, n; 10, 20, 100, 0.005, 1, 1)$ for one hundred times for generating the *Ex-PSO-A* designs, because the exchange algorithm seems inefficient under A-optimal criterion, as shown in Table 2.

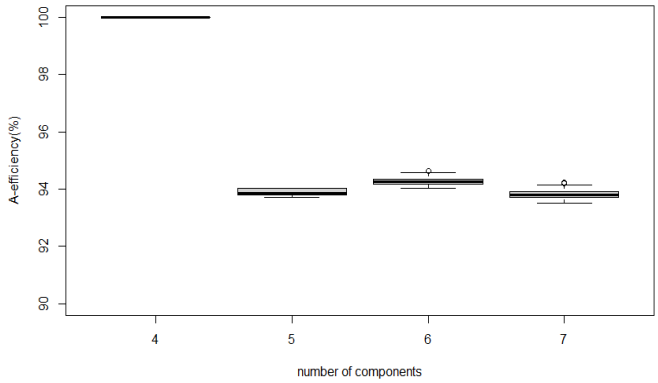


Figure 3: Boxplot of the A-efficiencies for the *Ex-PSO-A* designs with $m = 4, 5, 6, 7$ components and $n = m(m-1)$ generated by one hundred runs of Ex-PSO$(m, n; 10, 20, 100, 0.005, 1, 1)$.

Therefore, one hundred *Ex-PSO-A* designs with $m = 4, 5, 6, 7$ components and $n = m(m-1)$ runs are generated, and Figure 3 highlights that all *Ex-PSO-A* designs reach at least 93% of the efficiency of the full PWO design. For the cases with large m, the settings on maximum iterations, t_{ex} and t_{pso} may not be enough, but the *Ex-PSO* algorithm still returns approximate optimal PWO designs; see Tables 4-6 in the following part.

To illustrate the advantages of the obtained designs for fitting PWO model, we compare the *Ex-PSO-D* designs for 4 components and 12 runs with the optimal PWO design in Peng, Mukerjee, and Lin (2019).

Example 5.1. *The following is a* Ex-PSO-D *design with* 4 *components and* 12 *runs generated by Ex-PSO*$(4, 12; 10, 20, 100, 0.005, 1, 1)$.

Table 3. An Ex-PSO-D *design with* 4 *components and* 12 *runs*

Run	*Order-of-Addition*	$z_{12}, z_{13}, z_{14}, z_{23}, z_{24}, z_{34}$
1	1 4 2 3	1, 1, 1, 1, −1, −1
2	1 2 4 3	1, 1, 1, 1, 1, −1
3	1 3 2 4	1, 1, 1, −1, 1, 1
4	2 1 3 4	−1, 1, 1, 1, 1, 1
5	2 4 3 1	−1, −1, −1, 1, 1, −1
6	2 3 4 1	−1, −1, −1, 1, 1, 1
7	3 1 4 2	1, −1, 1, −1, −1, 1
8	3 2 1 4	−1, −1, 1, −1, 1, 1
9	3 4 1 2	1, −1, −1, −1, −1, 1
10	4 1 3 2	1, 1, −1, −1, −1, −1
11	4 2 1 3	−1, 1, −1, 1, −1, −1
12	4 3 2 1	−1, −1, −1, −1, −1, −1

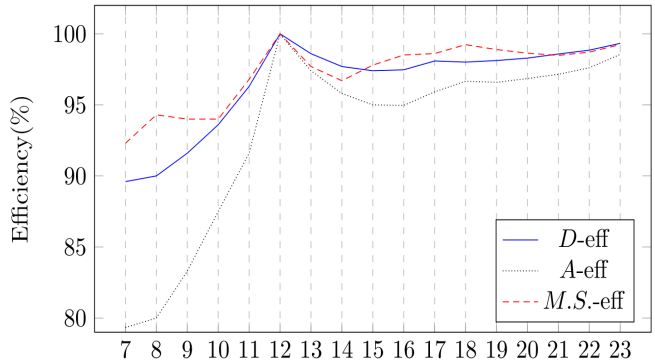


Figure 4: Relative efficiencies of *Ex-PSO* designs with $7 \leq n \leq 23$ compared with the full PWO design for the OofA experiment with 4 components.

The information matrix of this design under the first-order PWO model is

$$\widetilde{M} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1/3 & 1/3 & -1/3 & -1/3 & 0 \\ 0 & 1/3 & 1 & 1/3 & 1/3 & 0 & -1/3 \\ 0 & 1/3 & 1/3 & 1 & 0 & 1/3 & 1/3 \\ 0 & -1/3 & 1/3 & 0 & 1 & 1/3 & -1/3 \\ 0 & -1/3 & 0 & 1/3 & 1/3 & 1 & 1/3 \\ 0 & 0 & -1/3 & 1/3 & -1/3 & 1/3 & 1 \end{pmatrix}.$$

If rows are rearranged, this design is the same as the optimal PWO design with 4!/2! *runs constructed by Peng, Mukerjee, and Lin (2019). This design also features projective properties (Voelkel and Gallagher 2019). All 4 subsets of three components correspond to two-times-replicated three-component designs.*

Furthermore, for the OofA experiment with 4 components, we generated optimal PWO designs with 7 to 23 runs using the *Ex-PSO* algorithm. Figure 4 shows the efficiencies of these designs. Clearly, all the obtained designs with $n \geq 12$ reach at least 95% efficiency of the full PWO design, though with less than one fifth of the runs. Especially for the cases with $n = 12$, the design attains the same efficiency as the full PWO design.

Furthermore, we compare four types of fractional PWO designs, which are COA, and the corresponding designs obtained by the threshold accepting algorithm (Winker, Chen, and Lin 2020), the Federov's exchange algorithm (which iteratively optimizes a delta function of the x_i and x where x_i is in the design and x is not, see reference to section 3.3 in Fedorov 1972) and the *Ex-PSO* algorithm, denoted as D_{coa}, D_{ta}, D_{ex} and D_{ex-pso} respectively. D_{ta} is the best result obtained over repeated runs of threshold accepting algorithm with up to 10000000 iterations, D_{ex} is generated by the *optFederov* function (implemented in the *R* library *AlgDesign*) with $nRepeats = 5$, and D_{ex-pso} is the best result obtained over five repeated runs of the *Ex-PSO* algorithm with $t_{ex} = 20$ and $t_{pso} = 100$. The optimal PWO design constructed in Peng, Mukerjee, and Lin (2019) which

serves as a benchmark for evaluating fractional PWO designs is also listed here and denoted as D_{peng}. In addition, the new hybrid algorithm needs exhaustive search over the design space during the single-point exchange procedure, and it can be computational expensive if m is large. Hence, we only report designs with $n = m(m-1)/2+1, m(m-1), m!/r!(r = \lfloor m/2 \rfloor)$ where $4 \leq m \leq 7$. Nevertheless, given the tremendous growth in computational resources available, it is feasible to conduct the *Ex-PSO* algorithm for constructing designs with $m > 7$.

Tables 4-6 exhibit the values of $det(\widetilde{M})^{1/p}$, $tr(\widetilde{M}^{-1})$, $tr(\widetilde{M}^2)$, and D-, A- and $M.S.$-efficiency (in parentheses) for the corresponding designs. Note that the larger the value of $det(\widetilde{M})^{1/p}$ is, the better, while smaller values of $tr(\widetilde{M}^{-1})$ and $tr(\widetilde{M}^2)$ are better. For any number of components, D_{mp} is not unique and the corresponding $tr(\widetilde{M}^{-1})$ or $tr(\widetilde{M}^2)$ is by no means a fixed value. Hence, D_{mp} is not listed in Tables 5 and 6. From the tables, we can find that D_{ex-pso} reach a higher efficiency than the other types of designs under the PWO model in most cases. Further, we report the best PWO designs with $n = m(m-1)/2+1, m(m-1)$ and $4 \leq m \leq 7$ under the D-, A- and $M.S.$-optimal criteria in the supplementary material.

Table 4. Comparison of $det(\widetilde{M})^{1/p}$ and D-efficiency of PWO designs

m	n	D_{peng}	D_{coa}	D_{ta}	D_{ex}	D_{ex-pso}
4	7	-	-	0.6966(89.6%)	0.6966(89.6%)	0.6966(89.6%)
	12	0.7773(100%)	0.7064(90.9%)	0.7773(100%)	0.7773(100%)	0.7773(100%)
5	11	-	-	0.6379(90.3%)	0.6211(87.9%)	0.6379(90.3%)
	20	-	0.6354(89.9%)	0.6855(97.0%)	0.6840(96.8%)	0.6855(97.0%)
	60	0.7067(100%)	-	0.7067(100%)	0.7061(99.9%)	0.7067(100%)
6	16	-	-	0.5778(88.1%)	0.5612(85.6%)	0.6002(91.5%)
	30	-	0.5710(87.1%)	0.6344(96.7%)	0.6372(97.2%)	0.6381(97.3%)
	120	0.6558(100%)	-	0.6552(99.9%)	0.6555(99.95%)	0.6558(100%)
7	22	-	-	0.5016(81.2%)	0.5325(86.2%)	0.5409(87.6%)
	42	-	0.5800(93.9%)	0.5958(96.4%)	0.5996(97.0%)	0.5998(97.1%)
	840	0.6178(100%)	-	0.6178(100%)	0.6177(99.99%)	0.6178(100%)

Table 5. Comparison of $tr(\widetilde{M}^{-1})$ and A-efficiency of PWO designs

m	n	D_{peng}	D_{coa}	D_{ta}	D_{ex}	D_{ex-pso}
4	7	-	-	17.5000(67.4%)	-	14.8750(79.3%)
	12	11.8000(100%)	14.5000(81.4%)	11.8000(100%)	11.8000(100%)	11.8000(100%)
5	11	-	-	28.2898(74.2%)	-	26.4773(79.3%)
	20	-	26.0000(80.8%)	22.4550(93.5%)	22.3910(93.8%)	22.3311(94.0%)
	60	21.0000(100%)	-	21.0337(99.8%)	21.0000(100%)	21.0000(100%)
6	16	-	-	46.0558(72.0%)	-	40.8428(81.2%)
	30	-	45.3736(73.0%)	35.3203(93.8%)	35.0989(94.4%)	35.0144(94.7%)
	120	33.1429(100%)	-	33.2443(99.7%)	33.2023(99.8%)	33.1721(99.9%)
7	22	-	-	76.4729(63.1%)	-	72.4088(66.6%)
	42	-	57.1177(84.5%)	51.6845(93.4%)	51.0578(94.5%)	51.5024(93.7%)
	840	48.2500(100%)	-	48.2583(99.98%)	48.2738(99.95%)	48.2555(99.99%)

Note: D_{ex} with $n = m(m-1)/2+1$ is omitted because it reports an error of "singular design" when running the *optFederov* function from the *AlgDesign* package in *R*.

Table 6. Comparison of $tr(\widetilde{M}^{2})$ and $M.S.$-efficiency of PWO designs

m	n	D_{peng}	D_{coa}	D_{ta}	D_{ex-pso}
4	7	-	-	10.4694(92.3%)	10.4694(92.3%)
	12	9.6667(100%)	10.3333(93.6%)	9.6667(100%)	9.6667(100%)
5	11	-	-	18.5207(95.4%)	18.5207(95.4%)
	20	-	19.0000(93.0%)	18.0400(97.9%)	18.0000(98.2%)
	60	17.6667(100%)	-	17.6667 (100%)	17.6667(100%)
6	16	-	-	31.0000(94.6%)	30.9688(94.7%)
	30	-	31.3333(93.6%)	29.8756(98.2%)	29.8311(98.3%)
	120	29.3333(100%)	-	29.3556(99.92%)	29.3733(99.89%)
7	22	-	-	47.5702(95.3%)	47.7686(94.9%)
	42	-	45.8095(99.0%)	45.9048 (98.8%)	46.1905(98.1%)
	840	45.3333(100%)	-	45.3349(99.99%)	45.3368(99.99%)

We conclude this section with some numerical results on constructions of fractional PWO designs which have the same correlation structure as the full PWO design. Since these designs are 100% efficient under diverse design criteria including the D-, A-, $M.S.$-optimal criteria, we call them fully efficient PWO designs. Using the *Ex-PSO* algorithm, we find the following results.

Remark 5.2. *Removing $p(< m)$ components from a fully efficient PWO design with m components will result in a fully efficient PWO design with $m-p$ components.*

Table 7. Selected fully efficient PWO designs for $m = 4, 5, 6, 7$

$m = 4$	$m = 5$	$m = 6$		$m = 7$	
		runs 1-12	runs 13-24	runs 1-12	runs 13-24
1 2 4 3	1 2 3 5 4	1 2 5 4 6 3	4 2 1 3 6 5	1 2 3 7 4 6 5	4 5 2 6 7 1 3
1 3 4 2	1 4 3 5 2	1 2 5 4 6 3	4 2 5 3 6 1	1 5 6 3 2 7 4	4 6 3 7 1 2 5
1 3 2 4	1 5 3 2 4	1 3 4 2 5 6	4 6 3 2 5 1	1 6 5 7 4 2 3	4 7 1 3 6 5 2
2 1 4 3	2 4 3 1 5	1 3 6 2 5 4	5 1 2 4 3 6	1 7 4 3 2 5 6	5 2 4 3 6 1 7
2 3 1 4	2 5 1 4 3	1 4 6 5 2 3	5 4 3 2 1 6	2 5 4 1 7 6 3	6 1 2 4 5 7 3
2 3 4 1	3 1 4 2 5	2 3 4 5 1 6	5 2 1 6 3 4	2 7 6 3 1 5 4	6 4 2 1 3 7 5
3 1 4 2	3 2 4 5 1	2 6 1 5 3 4	5 3 6 2 1 4	3 2 1 6 4 7 5	6 5 1 3 4 7 2
3 2 4 1	3 5 4 2 1	2 6 4 3 1 5	5 6 4 1 2 3	3 2 6 5 7 4 1	6 7 2 3 4 5 1
4 1 2 3	4 2 1 5 3	3 1 2 6 4 5	6 2 1 4 3 5	3 4 1 5 7 2 6	7 4 6 5 3 2 1
4 2 1 3	4 5 1 2 3	3 1 5 6 4 2	6 2 5 3 4 1	3 5 1 4 6 2 7	7 2 1 5 3 4 6
4 3 1 2	5 2 3 1 4	3 5 2 4 6 1	6 3 5 1 4 2	3 5 7 6 2 1 4	7 5 1 2 6 4 3
4 3 2 1	5 4 3 1 2	4 1 5 6 3 2	6 4 5 1 3 2	4 2 5 3 7 1 6	7 5 3 6 4 1 2

Remark 5.3. *The fully efficient PWO designs exist for the cases (i)* $m = 4, 5$, $n = 12k(k \geq 1)$*; (ii)* $m = 6$, $n = 24k(k \geq 1)$*; and (iii)* $m = 7$, $n = 24$.

For saving space, some selected fully efficient PWO designs with minimized runs for $m = 4, 5, 6, 7$ are exhibited in Table 7, other fully efficient PWO designs and the MATLAB codes for the *Ex-PSO* algorithm are available upon request.

6. Concluding Remarks

For the OofA experiments, the study of the optimal fraction of the PWO design has received considerable attention in the literature. The fractional PWO design with the same correlation structure as the full PWO design is optimal under diverse design criteria but exists only for some fixed run sizes, such as $m!/r!(2 \leq r \leq m)$ runs. Theoretical constructions on optimal PWO designs are also heavily constrained by the run size. In this paper, we present a flexible and effective searching algorithm, the *Ex-PSO* algorithm. Even though the candidate fractional PWO designs are extremely massive, this algorithm generates high efficient designs with only one hundred iterations. Moreover, it's an interesting but difficult problem to obtain more general theoretical results which cover Remark 4 as special cases. While Remark 3 gives a fresh insight into constructions of the fully efficient PWO design with m components basing on the fully efficient PWO design with $m - 1$ components. To that effect, more theoretical results on the fully efficient PWO designs for general m will be studied in our future work.

It is worth noting that the *Ex-PSO* designs are possibly to an optimal PWO designs given the tremendous growth in computational resources available, thus it provides instructions for exploring theoretical results on optimal PWO designs. In addition, the *Ex-PSO* algorithm applies not only to PWO design but also to any type of design with finite candidate points. Therefore, this algorithm has many potential applications, such as constructing optimal designs for an alternative model of the OofA experiment or other kinds of experiments, and there are still many issues for further

study.

Acknowledgements

The authors thank the editor and two referees for their valuable comments and suggestions. This work was supported by the National Natural Science Foundation of China (grant nos. 11971098, 12101258 and 12131001), National Key Research and Development Program of China (No. 2020YFA0714102) and Education Department Science and Technology Project of Jilin Province under Grant JJKH20220152KJ.

References

[1] Atwood C. L. 1969. "Optimal and efficient designs of experiments." *The Annals of Mathematical Statistics* 40 (5): 1570-1602. doi:10.1214/aoms/1177697374.

[2] Chen P. Y., Chen R. B., and Wang W. K. 2022. *Particle swarm optimization for searching efficient experimental designs: A review.* Wiley Interdisciplinary Reviews: Computational Statistics.

[3] Chen R. B., Hsu Y. W., Hung Y., and Wang W. C. 2014. "Discrete particle swarm optimization for constructing uniform design on irregular regions." *Computational Statistics & Data Analysis* 72: 282-297. doi:10.1016/j.csda.2013.10.015.

[4] Eberhart R. C., and Kennedy J. 2002. "A new optimizer using particle swarm theory," MHS'95. *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, 39-43. doi:10.1109/MHS.1995.494215.

[5] Fedorov V. V. 1972. *Theory of optimal experiments.* Translated and edited by Studden, W. J. and Klimko E. M. New York.

[6] Fisher R. A. 1937 *The Design of Experiments.* London: Edinburgh.

[7] Fuleki T., Francis F. J. 1968. "Quantitative methods for anthocyanins." *Journal of Food Science* 33: 266-274.

[8] Jourdain L. S., Schmitt C., Leser M. E., Murray B. S., and Dickinson E. 2009. "Mixed layers of sodium caseinate+dextran sulfate: influence of order of addition to oil-water interface." *Langmuir* 25: 10026-10037. doi:10.1021/la900919w.

[9] Karim M., Mccormick K., and Kappagoda C. T. 2000. "Effects of cocoa extracts on endothelium-dependent relaxation." *The Journal of Nutrition* 130: 2105S-2108S. doi:10.1093/jn/130.8.2105S.

[10] Lin D. K. J., and Peng J. Y. 2019. Order-of-addition Experiments: A review and some new thoughts, *Quality Engineering* 31: 49-59. doi:10.1080/08982112.2018.1548021.

[11] Mak S., and Joseph V. J. 2018. "Minimax and minimax projection designs using clustering." *Journal of Computational and Graphical Statistics* 27: 166-178. doi:10.1080/10618600.2017.1302881.

[12] Mee R. W. 2020. "Order-of-Addition Modeling." *Statistica Sinica* 30 (3): 1543-1559. doi:stable/26968940.

[13] Mitchell T. J. 2000. "An Algorithm for the construction of 'D-Optimal' experimental designs." *Technometrics* 42 (2): 48-54. doi:10.1080/00401706.1974.10489175.

[14] Nguyen N. K., and Miller A. J. 1992. "A review of some exchange algorithms for constructing discrete D-optimal designs." *Computational Statistics & Data Analysis* 14: 489-498. doi:10.1016/0167-9473(92)90064-M.

[15] Olsen G. J., Matsuda H., Hagstrom R., and Overbeek R. 1994. "Fastdnaml: a tool for construction of phylogenetic trees of dna sequences using maximum likelihood." *Computer Applications in the Biosciences: CABIOS* 10: 41-48. doi:10.1093/bioinformatics/10.1.41.

[16] Peng J. Y., Mukerjee R., and Lin D. K. J. 2019. "Design of order-of-addition experiments." *Biometrika* 106 (3): 683-694. doi:10.1093/BIOMET/ASZ025.

[17] Phoa F. K. H., Chen R. B., Wang W. C., and Wong W. 2016. "Optimizing two-level supersaturated designs using swarm intelligence techniques." *Technometrics* 58: 43-49. doi:10.1080/00401706.2014.981346.

[18] Van Nostrand R. C. 1995. *Design of experiments where the order-of-addition is important.* ASA Proceedings of the Section on Physical and Engineering Sciences, Alexandria, Virginia, 155-160.

[19] Voelkel J. G. 2019. "The Design of order-of-addition experiments." *Journal of Quality Technology* 51 (3): 230-241. doi:10.1080/00224065.2019.1569958.

[20] Voelkel J. G., and Gallagher K. P. 2019. "The design and analysis of order-of-addition experiments: An introduction and case study." *Quality Engineering* 31(4): 1-12. doi:10.1080/08982112.2019.1578374.

[21] Winker P., Chen J. B., and Lin D. K. J. 2020. "The construction of optimal design for order-of-addition experiment via threshold accepting." Chap 6 in: *Contemporary Experimental Design, Multi-variate Analysis and Data Mining.* Switzerland: Cham.

[22] Yang J. F., Sun F. S., and Xu H. Q. 2021. "A component-position model, analysis and design for order-of-addition experiments." *Technometrics* 63 (2): 212-224. doi:10.1080/00401706.2020.1764394.

[23] Zhao Y. N., Lin D. K. J., and Liu M. Q. 2021. "Designs for order of addition experiments." *Journal of Applied Statistics* 48 (8), 1475-1495. doi:10.1080/02664763.2020.1801607.

[24] Zhao Y. N., Lin D. K. J., and Liu M. Q. 2022. "Optimal designs for order-of-addition experiments." *Computational Statistics & Data Analysis* 165. doi:10.1016/j.csda.2021.107320.

Appendix A. The proof of Theorem 3.1

To prove Theorem 3.1, the following two lemmas are useful.

Lemma A.1. *For a nonsingular matrix M,*

(1) $M + xx'$ is nonsingular, and $(M + xx')^{-1} = M^{-1} - wuu'$, where $w = 1/(1 + x'M^{-1}x)$, $u = M^{-1}x$;

(2) if $M_x - x_i x_i'$ is nonsingular, then $x_i'M^{-1}x_i \neq 1$ and $(M_x - x_i x_i')^{-1} = M_x^{-1} + w_i u_i u_i'$, where $w_i = 1/(1 - x_i'M^{-1}x_i)$, $u_i = M_x^{-1}x_i$.

The proof of this lemma is straightforward according to matrix theory and is thus omitted.

Lemma A.2. *Let M be a nonsingular matrix and $M_x = M + xx'$; we have $x'M_x^{-1}x = \frac{x'M^{-1}x}{1+x'M^{-1}x}$ and $x'M_x^{-2}x = \frac{x'M^{-2}x}{(1+x'M^{-1}x)^2}$.*

Proof. According to Lemma A.1, we have

$$\begin{aligned} x'M_x^{-1}x &= x'M^{-1}x - wx'uu'x \\ &= x'M^{-1}x - \frac{(x'M^{-1}x)^2}{1 + x'M^{-1}x} \\ &= \frac{x'M^{-1}x}{1 + x'M^{-1}x}, \end{aligned}$$

and

$$\begin{aligned} x'M_x^{-2}x &= x'(M^{-1} - wuu')^2x \\ &= x'M^{-2}x - 2wx'M^{-1}uu'x + w^2x'(uu')^2x \\ &= x'M^{-2}x - \frac{2x'M^{-2}xx'M^{-1}x}{1 + x'M^{-1}x} + \frac{x'(M^{-1}xx'M^{-1})^2x}{(1 + x'M^{-1}x)^2} \\ &= x'M^{-2}x[1 - \frac{2x'M^{-1}x}{1 + x'M^{-1}x} + \frac{(x'M^{-1}x)^2}{(1 + x'M^{-1}x)^2}] \\ &= \frac{x'M^{-2}x}{(1 + x'M^{-1}x)^2}. \end{aligned}$$

□

Proof of Theorem 3.1

Since Fedorov's exchange algorithm has proved this result for the case with $\phi(D) = det(M)$, $u_D(x)$ and $v_D(x_i)$ defined as Equation (4), hence we only prove this result for the other two cases.

First, we prove that the design generated by single-point exchange procedure leads to no increase in $tr(M^{-1})$.

Let X be the model matrix of the current design and denote $M = X'X$, which is updated as $M_x - x_i x_i'$ after exchanging x for x_i according to the single exchange procedure. The following delta function evaluates the multiple changes from $tr(M^{-1})$ to $tr((M_x - x_i x_i')^{-1})$:

$$\begin{aligned}
\triangle(x_i, x) &= \frac{tr((M_x - x_i x_i')^{-1})}{tr(M^{-1})} \\
&= \frac{tr(M^{-1}) - tr(wuu') + tr(w_i u_i u_i')}{tr(M^{-1})} \\
&= 1 - \frac{tr(wu'u) - tr(w_i u_i' u_i)}{tr(M^{-1})} \\
&= 1 - \frac{1}{tr(M^{-1})}\left[\frac{x'M^{-2}x}{1 + x'M^{-1}x} - \frac{x_i' M_x^{-2} x_i}{1 - x_i' M_x^{-1} x_i}\right]. \qquad \text{(A.1)}
\end{aligned}$$

Based on step (2) of this procedure, we know that $\frac{x_i' M_x^{-2} x_i}{1 - x_i' M_x^{-1} x_i} \leq \frac{x' M_x^{-2} x}{1 - x' M_x^{-1} x}$. In the combination of Lemma A.2, we obtain

$$\begin{aligned}
\frac{x'M^{-2}x}{1 + x'M^{-1}x} - \frac{x_i' M_x^{-2} x_i}{1 - x_i' M_x^{-1} x_i} &\geq \frac{x'M^{-2}x}{1 + x'M^{-1}x} - \frac{x' M_x^{-2} x}{1 - x' M_x^{-1} x} \\
&= \frac{x'M^{-2}x}{1 + x'M^{-1}x} - \frac{\frac{x'M^{-2}x}{(1+x'M^{-1}x)^2}}{1 - \frac{x'M^{-1}x}{1+x'M^{-1}x}} \\
&= \frac{x'M^{-2}x}{1 + x'M^{-1}x} - \frac{x'M^{-2}x}{1 + x'M^{-1}x} \\
&= 0.
\end{aligned}$$

Thus, $\triangle(x_i, x) \leq 1$ is obtained, which means $tr(M^{-1})$ does not increase in the single-point exchange procedure.

Second, we prove that the design generated in single-point procedure leads to no increase in $tr(M^2)$.

To calculate the multiple exchange on the $tr(M^2)$ during each iteration of the single-point

exchange procedure, we define a delta function $\triangle(x_i, x)$ as follows:

$$\begin{aligned}
\triangle(x_i, x) &= \frac{tr((M_x - x_i x_i')^2)}{tr(M^2)} \\
&= \frac{tr(M_x^2) - 2x_i' M_x x_i + p^2}{tr(M^2)} \\
&= \frac{tr(M^2) + 2x'Mx - 2x_i' M_x x_i + 2p^2}{tr(M^2)} \\
&= 1 + \frac{2x'Mx - 2x_i' M_x x_i + 2p^2}{tr(M^2)}. \qquad \text{(A.2)}
\end{aligned}$$

By step (2) of this procedure, we have $x'M_x x \leq x_i' M_x x_i$ and

$$\begin{aligned}
x'Mx - x_i' M_x x_i &\leq x'Mx - x'M_x x \\
&\leq x'Mx - x'(M + xx')x \\
&\leq -p^2. \qquad \text{(A.3)}
\end{aligned}$$

Thus, we obtain $\triangle(x_i, x) \leq 1$ from (A.2) and (A.3).

Theorem 3.1 is proved.

Appendix B. Best PWO designs under the *D*-optimal criterion

Table B.1 *D*-Optimal PWO designs for $m = 4$

7 runs				12 runs							
				runs 1-7				runs 8-12			
1	2	3	4	1	2	4	3	3	2	1	4
1	3	4	2	1	2	3	4	3	2	4	1
2	1	4	3	1	3	4	2	4	1	3	2
3	1	2	4	2	1	4	3	4	2	1	3
3	2	4	1	2	3	4	1	4	2	3	1
4	1	3	2	3	1	4	2	4	3	1	2
4	2	3	1								

Table B.2 *D*-Optimal PWO designs for $m = 5$

11 runs					20 runs									
					runs 1-10					runs 11-20				
1	5	3	4	2	1	5	2	3	4	3	5	4	1	2
2	5	3	4	1	1	2	5	4	3	3	4	2	1	5
2	3	1	4	5	1	3	4	2	5	4	1	5	3	2
2	4	1	3	5	1	4	2	3	5	4	3	1	5	2
3	2	5	1	4	2	1	4	3	5	4	3	5	1	2
3	4	5	1	2	2	5	4	1	3	4	5	3	2	1
4	2	5	1	3	2	3	4	5	1	5	1	3	2	4
4	3	1	2	5	2	4	5	3	1	5	2	3	1	4
4	5	3	2	1	3	2	1	5	4	5	3	1	4	2
5	1	2	4	3	3	5	2	1	4	5	4	2	1	3
5	4	2	3	1										

Table B.3 D-Optimal PWO designs for $m = 6$

16 runs						30 runs											
						runs 1-15						runs 16-30					
1	6	5	2	4	3	1	6	2	5	3	4	4	5	1	6	2	3
1	2	4	5	3	6	1	2	6	3	5	4	4	5	2	6	1	3
1	5	3	6	4	2	1	3	4	2	5	6	4	5	3	1	2	6
2	1	3	5	4	6	1	4	6	3	5	2	5	2	1	3	6	4
2	3	6	5	4	1	2	1	5	4	6	3	5	2	3	1	4	6
3	1	4	2	6	5	2	6	4	1	5	3	5	4	1	3	6	2
3	2	5	1	6	4	2	3	5	4	6	1	5	4	3	6	1	2
3	4	5	1	6	2	3	1	5	2	6	4	5	6	3	1	2	4
4	1	3	5	2	6	3	1	6	4	2	5	5	6	3	4	2	1
4	2	5	1	6	3	3	2	6	5	1	4	6	1	5	4	2	3
4	3	6	5	2	1	3	2	4	1	6	5	6	1	2	3	4	5
5	2	6	3	1	4	3	4	6	1	5	2	6	2	1	4	5	3
5	4	6	3	1	2	3	5	6	2	1	4	6	2	4	3	5	1
6	1	4	2	3	5	4	2	1	3	5	6	6	3	5	4	2	1
6	3	2	4	1	5	4	2	6	5	3	1	6	5	1	3	2	4
6	5	1	3	2	4												

Table B.4 D-Optimal PWO designs for $m = 7$

22 runs							42 runs												
							runs 1-21							runs 22-42					
1	7	5	2	6	4	3	1	7	6	5	2	3	4	4	3	7	5	6	2	1
1	4	3	6	5	7	2	1	4	7	2	3	6	5	4	7	2	1	5	6	3
2	5	6	1	3	4	7	1	4	3	2	7	6	5	4	6	2	1	7	5	3
2	6	3	4	7	1	5	1	5	6	4	2	7	3	4	6	5	3	7	1	2
3	1	2	5	6	7	4	1	6	2	4	5	3	7	5	1	6	4	7	3	2
3	2	7	4	6	5	1	2	1	4	3	6	7	5	5	2	1	7	4	6	3
3	7	5	6	1	4	2	2	3	7	1	6	4	5	5	2	3	6	4	1	7
3	5	4	1	7	6	2	2	3	5	4	7	6	1	5	3	1	4	6	7	2
4	2	1	7	5	3	6	2	5	7	1	3	4	6	5	7	2	6	1	4	3
4	5	6	1	2	7	3	2	5	4	7	3	6	1	6	1	7	2	3	4	5
5	2	7	1	6	3	4	2	5	6	1	3	7	4	6	1	5	3	4	2	7
5	3	6	7	2	4	1	2	6	7	5	4	3	1	6	3	2	4	7	5	1
5	4	6	3	2	7	1	2	6	4	7	3	5	1	6	3	4	5	1	2	7
5	7	4	2	3	6	1	3	1	7	2	5	6	4	6	7	1	3	5	2	4
6	1	7	5	3	4	2	3	1	5	6	2	7	4	6	7	1	4	2	5	3
6	2	1	3	7	5	4	3	2	4	1	6	5	7	7	3	1	4	5	2	6
6	2	5	4	7	3	1	3	7	5	4	1	2	6	7	3	6	4	2	5	1
6	4	1	3	2	5	7	3	6	5	7	4	2	1	7	4	1	6	2	3	5
6	7	4	3	2	5	1	4	1	3	2	5	6	7	7	4	5	3	6	1	2
7	1	3	2	5	4	6	4	1	5	7	3	2	6	7	6	3	2	1	5	4
7	1	4	6	2	5	3	4	2	6	3	1	5	7	7	6	5	4	1	3	2
7	6	3	2	1	4	5													

Appendix C. Best PWO designs under the *A*-optimal criterion

Table C.1 *A*-Optimal PWO designs for $m = 4$

7 runs				12 runs							
				runs 1-7				runs 8-12			
1	3	4	2	1	4	3	2	3	1	2	4
2	1	4	3	1	2	3	4	3	2	4	1
2	3	1	4	2	1	3	4	3	4	2	1
3	1	2	4	2	1	4	3	4	1	3	2
3	2	4	1	2	4	3	1	4	1	2	3
4	1	2	3	3	1	4	2	4	2	3	1
4	3	2	1								

Table C.2 *A*-Optimal PWO designs for $m = 5$

11 runs					20 runs									
					runs 1-10					runs 11-20				
1	4	2	3	5	1	5	3	4	2	3	5	1	4	2
2	1	5	4	3	1	2	5	4	3	3	5	2	4	1
2	1	3	4	5	1	3	2	4	5	4	1	5	2	3
2	3	5	4	1	2	1	4	3	5	4	2	1	3	5
2	4	5	1	3	2	5	4	3	1	4	2	5	1	3
3	4	1	2	5	2	3	5	4	1	4	3	1	5	2
4	1	3	5	2	2	3	4	1	5	4	5	3	2	1
4	3	2	5	1	3	1	4	2	5	5	1	2	4	3
4	5	2	3	1	3	2	1	5	4	5	2	3	1	4
5	1	3	4	2	3	2	4	5	1	5	4	1	3	2
5	3	1	2	4										

Table C.3 A-Optimal PWO designs for $m = 6$

16 runs						30 runs											
						runs 1-15						runs 16-30					
2	1	5	6	3	4	1	3	2	6	4	5	4	1	5	2	3	6
3	1	2	4	5	6	1	3	5	6	2	4	4	2	1	3	5	6
3	1	6	4	5	2	1	4	2	6	3	5	4	3	2	5	6	1
3	5	2	4	1	6	1	4	3	6	2	5	4	6	1	2	3	5
3	5	6	4	1	2	2	1	6	4	5	3	4	5	1	6	3	2
4	2	3	6	5	1	2	1	6	3	5	4	5	2	3	1	4	6
4	3	6	2	1	5	2	1	4	5	6	3	5	2	4	6	1	3
4	5	6	2	3	1	2	3	5	1	6	4	5	3	4	1	2	6
5	1	6	4	3	2	2	5	3	6	4	1	5	6	1	3	4	2
5	1	3	4	6	2	2	5	4	3	1	6	6	1	4	2	5	3
5	3	2	6	1	4	3	1	2	5	4	6	6	2	1	5	3	4
6	1	5	2	4	3	3	1	4	5	6	2	6	2	3	4	5	1
6	1	3	4	5	2	3	2	4	6	1	5	6	4	3	1	5	2
6	2	3	5	1	4	3	4	6	5	2	1	6	5	1	2	4	3
6	2	4	5	1	3	3	5	2	6	1	4	6	5	4	3	2	1
6	5	3	4	1	2												

Table C.4 A-Optimal PWO designs for $m = 7$

22 runs							42 runs												
							runs 1-21							runs 22-42					
1	3	7	4	2	6	5	1	2	7	3	5	4	6	4	7	3	6	1	5	2
1	5	4	6	3	7	2	1	3	7	6	5	2	4	4	7	6	1	2	5	3
1	6	7	4	5	3	2	1	5	4	6	3	7	2	4	5	2	1	3	6	7
1	6	3	4	5	7	2	1	6	2	5	3	4	7	5	1	7	4	3	6	2
2	7	6	1	4	3	5	1	6	4	7	3	2	5	5	3	6	4	1	7	2
2	4	5	6	1	3	7	1	6	4	5	2	7	3	5	7	3	2	4	1	6
2	6	5	4	3	1	7	2	1	4	5	7	6	3	5	7	3	6	1	4	2
3	6	1	5	7	2	4	2	7	6	5	1	4	3	5	6	4	3	1	2	7
3	6	2	5	1	4	7	2	4	3	5	6	7	1	6	1	7	3	5	2	4
3	6	4	2	7	1	5	2	4	5	6	1	3	7	6	1	3	4	5	2	7
4	1	6	2	7	3	5	2	6	5	4	7	1	3	6	1	5	3	7	4	2
4	1	5	2	7	3	6	3	1	2	4	6	7	5	6	2	3	5	1	4	7
4	2	3	5	6	7	1	3	1	7	2	6	4	5	6	2	4	1	7	5	3
4	3	1	7	5	6	2	3	2	7	1	5	4	6	6	3	5	2	7	4	1
4	7	6	5	3	1	2	3	4	2	1	7	5	6	6	7	4	3	1	2	5
5	3	2	7	4	1	6	3	4	5	7	1	2	6	6	7	5	2	3	1	4
5	6	3	1	4	2	7	3	6	2	7	1	4	5	7	1	5	2	6	4	3
7	1	6	2	4	5	3	4	2	6	3	1	5	7	7	2	1	3	5	6	4
7	1	3	2	5	6	4	4	2	6	7	5	1	3	7	4	5	1	6	2	3
7	5	2	3	1	4	6	4	2	3	1	6	5	7	7	5	2	6	3	1	4
7	6	3	4	2	1	5	4	3	7	6	5	2	1	7	6	3	4	2	5	1
7	6	5	4	2	1	3													

Appendix D. Best PWO designs under the $M.S.$-optimal criterion

Table D.1 $M.S.$-Optimal PWO designs for $m = 4$

7 runs				12 runs							
				runs 1-7				runs 8-12			
1	2	4	3	1	4	3	2	2	3	1	4
2	1	3	4	1	4	2	3	3	2	1	4
2	4	3	1	1	2	3	4	3	4	1	2
3	1	4	2	1	3	2	4	3	4	2	1
3	2	4	1	2	4	1	3	4	2	1	3
4	1	3	2	2	4	3	1	4	3	1	2
4	2	1	3								

Table D.2 $M.S.$-Optimal PWO designs for $m = 5$

11 runs					20 runs									
					runs 1-10					runs 11-20				
1	5	2	4	3	1	5	3	4	2	3	5	2	1	4
1	5	3	4	2	1	2	5	4	3	4	1	3	2	5
1	3	2	4	5	1	3	4	5	2	4	2	5	1	3
2	5	1	4	3	2	1	4	3	5	4	2	3	1	5
2	5	3	4	1	2	1	5	3	4	4	3	1	5	2
2	3	1	4	5	2	3	1	4	5	4	5	1	2	3
3	5	1	4	2	2	3	4	5	1	4	5	3	2	1
3	5	2	4	1	3	1	4	2	5	5	1	2	4	3
4	1	2	3	5	3	2	5	4	1	5	3	2	4	1
4	3	2	1	5	3	5	1	4	2	5	4	2	3	1
4	5	2	1	3										

Table D.3 *M.S.*-Optimal PWO designs for $m = 6$

16 runs						30 runs											
						runs 1-15						runs 16-30					
1	2	6	4	5	3	1	4	2	3	5	6	4	2	1	6	3	5
1	3	4	6	5	2	1	4	3	2	6	5	4	5	6	2	3	1
1	5	2	4	3	6	1	4	5	3	6	2	5	1	6	3	4	2
2	6	3	4	5	1	2	1	6	5	3	4	5	1	2	6	4	3
2	5	1	3	4	6	2	6	4	5	1	3	5	2	1	3	4	6
3	2	6	1	5	4	2	3	5	6	1	4	5	3	1	2	4	6
3	4	1	2	5	6	2	3	4	1	5	6	5	3	6	4	2	1
3	5	1	6	4	2	2	4	6	5	3	1	5	4	2	3	6	1
4	1	3	6	2	5	3	1	6	5	2	4	5	4	6	1	3	2
4	2	3	5	6	1	3	2	5	4	1	6	6	1	2	3	4	5
4	6	5	3	1	2	3	2	6	4	1	5	6	2	1	5	4	3
5	3	6	2	4	1	3	4	6	2	5	1	6	3	1	4	2	5
5	4	2	1	6	3	3	4	6	5	1	2	6	3	1	5	2	4
5	6	1	4	3	2	4	1	3	5	2	6	6	4	3	5	2	1
6	1	2	3	5	4	4	1	6	2	5	3	6	5	4	1	2	3
6	4	2	1	5	3												

Table D.4 *M.S.*-Optimal PWO designs for $m = 7$

22 runs							42 runs												
							runs 1-21							runs 22-42					
1	2	3	6	7	4	5	1	7	2	6	3	4	5	4	2	6	7	3	5	1
1	2	5	7	6	3	4	1	7	2	4	5	6	3	4	3	5	1	7	2	6
1	4	3	6	2	5	7	1	7	5	3	2	6	4	4	7	5	3	2	1	6
2	1	6	5	4	3	7	1	3	2	5	7	4	6	4	5	3	6	2	1	7
2	5	4	6	3	1	7	1	3	4	5	6	2	7	4	5	7	6	2	1	3
3	4	1	7	5	6	2	1	4	3	2	6	7	5	4	6	3	7	1	5	2
3	4	2	7	6	5	1	1	5	7	4	2	3	6	5	2	1	6	4	3	7
4	5	2	6	7	1	3	1	6	5	2	4	7	3	5	3	2	4	7	6	1
4	5	3	1	7	2	6	2	7	5	1	3	4	6	5	4	1	6	3	7	2
4	6	1	7	2	5	3	2	3	4	1	6	5	7	5	7	4	2	3	6	1
5	2	1	7	3	4	6	2	3	5	6	4	7	1	5	6	7	3	1	2	4
5	3	6	1	7	4	2	2	4	7	6	1	5	3	6	1	4	2	5	3	7
5	3	6	2	4	7	1	2	4	3	7	1	6	5	6	1	5	3	4	7	2
6	1	4	5	7	3	2	2	5	3	7	1	4	6	6	2	5	1	7	4	3
6	3	2	1	7	4	5	2	6	3	1	7	5	4	6	5	3	4	1	2	7
6	3	5	7	1	2	4	3	1	4	6	7	2	5	6	7	2	5	4	1	3
6	7	5	4	2	3	1	3	2	6	4	5	1	7	6	7	4	5	1	2	3
7	1	5	4	6	3	2	3	7	6	4	2	5	1	7	1	3	6	5	4	2
7	2	4	3	1	5	6	3	7	5	6	1	2	4	7	4	6	1	3	2	5
7	3	2	6	1	5	4	3	6	7	2	1	4	5	7	5	3	4	1	2	6
7	5	1	3	2	4	6	4	2	1	7	5	6	3	7	6	3	5	2	1	4
7	6	2	4	5	1	3													